Started on	Tuesday, 28 May 2024, 3:18 PM
State	Finished
Completed on	Tuesday, 28 May 2024, 4:21 PM
Time taken	1 hour 2 mins
Grade	**430.00** out of 450.00 (**96**%)

Question **1**

Correct

Mark 150.00 out of 150.00

Time limit	1 s
Memory limit	64 MB

Tuan Haka ingin membuat parser nomor telepon dan email sendiri. Namun, dia ingin menerapkan konsep exception agar parsernya bekerja dengan aman. Bantulah Tuan Haka membuat parser nomor telepon dan email dengan menerapkan exception.

File pendukung dapat diunduh di sini.

Java 8

DataParser.java

Score: 150

Blackbox

Score: 150

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.06 sec, 28.35 MB
2	10	Accepted	0.57 sec, 28.55 MB
3	10	Accepted	0.18 sec, 28.44 MB
4	10	Accepted	0.13 sec, 27.91 MB
5	10	Accepted	0.79 sec, 28.63 MB
6	10	Accepted	0.62 sec, 26.25 MB
7	10	Accepted	0.40 sec, 28.11 MB
8	10	Accepted	0.26 sec, 30.23 MB
9	10	Accepted	0.14 sec, 28.34 MB
10	10	Accepted	0.07 sec, 28.83 MB
11	10	Accepted	0.23 sec, 28.08 MB
12	10	Accepted	0.62 sec, 28.02 MB
13	10	Accepted	0.80 sec, 28.98 MB
14	20	Accepted	0.56 sec, 28.58 MB

Question **2**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Tuan Haka ingin membuat sebuah service yang memberikan notifikasi ke banyak orang sekaligus. Selain itu service ini juga bisa memberikan notifikasi melalui email, sms, dan push notification. Tuan Haka ingin membuat service ini dengan menggunakan design pattern. Oleh karena itu, bantu Tuan Haka membuat service tersebut dengan design pattern Observer dan Strategy.

File-file pendukung dapat diunduh dari sini

Implementasikanlah kelas kelas berikut
- EmailMessageSender
- SMSMessageSender
- PushNotifMessageSender
- Service
- NotificationManager

Kumpulkan file-file tersebut menjadi sebuah zip bernama designPattern.zip

Urutan pengiriman pesan adalah email, push notif, dan sms

Java 8

designPattern.zip

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	20	Accepted	0.06 sec, 28.28 MB
2	20	Accepted	0.06 sec, 28.96 MB
3	20	Accepted	0.06 sec, 28.55 MB
4	20	Accepted	0.07 sec, 28.41 MB
5	20	Accepted	0.07 sec, 29.74 MB

Question **3**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Tuan Aditya adalah seorang techbro yang sangat masbro. Tuan Aditya membuat sebuah startup berbasis artifical intelligence yang mengklaim dirinya sangat canggih, bisa menyelesaikan rangkaian teka teki rumit milik Tuan Prim.

Teka teki Tuan Prim terdiri atas sejumlah bagian **piece**. Setiap piece memiliki satu bagian jawaban berupa sebuah huruf. Jawaban teka teki merupakan perpaduan jawaban seluruh piece yang diurutkan sesuai abjad, membentuk sederet kode seperti "AAGVXYZ".

Sebagai contoh, misal terdapat 3 piece. Piece pertama dipilih secara acak, dan ternyata jawabannya "B". Piece kedua dipilih secara acak, jawabannya "C". Piece terakhir jawabannya "B". Maka jawaban teka teki adalah "BBC".

Meski demikian, di balik itu produk yang dibuat Tuan Aditya sebenarnya hanya melakukan panggilan berkali-kali ke GPT (**Gpt.java**) untuk menyelesaikan bagian-bagian kecil dari teka teki dengan memasukkan kalimat tiap piece tersebut ke GPT. Bantulah Tuan Aditya membuat kode yang efektif untuk produknya tersebut, agar tidak ketahuan dan dikatai scam!

Unggah implementasi **Solver.java** dan **Scraper.java** dalam 1 zip. **Nama file zip bebas.**

Java 8

gipiti.zip

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	30	Accepted	0.72 sec, 28.46 MB
2	30	Accepted	0.67 sec, 28.35 MB
3	40	Accepted	0.64 sec, 28.40 MB

Question **4**

Partially correct

Mark 80.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Tuan Akbar mempunyai program yang akan menangani HTTP request. Tapi Tuan Akbar agak sok ngide karena itu Tuan Akbar ingin membuat handler-nya sendiri menggunakan decorator. Salah satu decorator yang digunakan adalah **HttpVar.java** yang akan digunakan seperti ini:

```
class SebuahHandler {
  @HttpVar(type = HttpVar.Type.Query, name = "q")
  private String q;

  private String token;

  // atribut lainnya...

  @HttpVar(type = HttpVar.Type.Header, name = "Authorization")
  public void initToken(String auth) {
    // implementasi set token dari header auth
  }

  // method-method handler...
}
```

Decorator @**HttpVar** dapat digunakan pada atribut dan method. Decorator ini akan digunakan oleh method **load()** pada **HttpVarLoader.java** untuk meng-inject nilai variabel HTTP yang dapat berupa variabel header, cookie, query, maupun body.

Contohnya, misalkan method **HttpVarLoader.load()** dipanggil dengan parameter sebuah objek SebuahHandler **h** (merujuk pada script di atas), maka:

- **h.q** akan diinject dengan nilai variabel HTTP query dengan nama/key "q"
- **h.initToken** akan dipanggil dengan parameter auth berupa nilai variabel HTTP header dengan nama/key "Authorization"

Key variabel HTTP case-sensitive sesuai dengan behavior HashMap.

Kumpulkan file **HttpVarLoader.java**.

Hint: Pada kelas **Field** dan **Method** terdapat method berikut:

- **isAnnotationPresent()** untuk memastikan apakah suatu annotation class dideklarasikan pada Field/Method tersebut
- **getAnnotation()** untuk mendapatkan objek annotation tertentu pada Field/Method tersebut

Java 8

HttpVarLoader.java

Score: 80

Blackbox

Score: 80

Verdict: Wrong answer

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.07 sec, 29.06 MB
2	0	Wrong answer	0.08 sec, 29.16 MB
3	10	Accepted	0.08 sec, 28.53 MB
4	10	Accepted	0.08 sec, 28.98 MB
5	10	Accepted	0.07 sec, 28.69 MB

No	Score	Verdict	Description
6	0	Wrong answer	0.08 sec, 28.49 MB
7	10	Accepted	0.07 sec, 28.54 MB
8	10	Accepted	0.07 sec, 28.18 MB
9	10	Accepted	0.07 sec, 28.64 MB
10	10	Accepted	0.06 sec, 28.02 MB

◄ Praktikum 6 & 7

Jump to...